Solving extremely ill-posed structures

Lars Beex Marc Duflot[b] Laurent Adam[b]

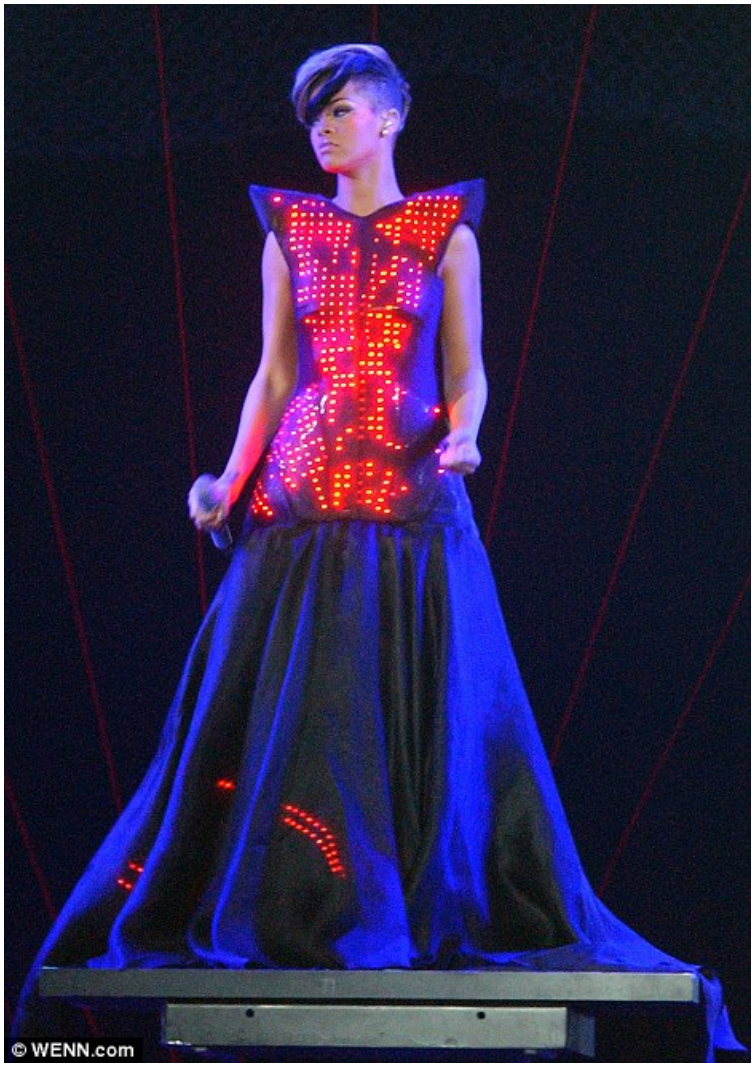
[b]e-Xstream Engineering

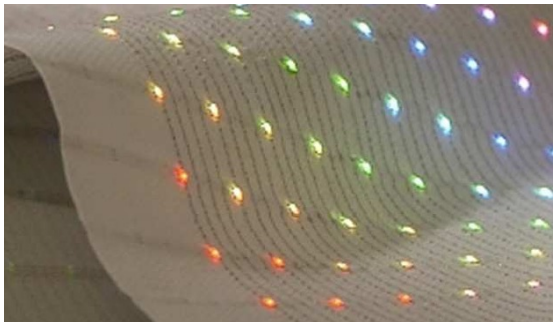
RUES | RESEARCH UNIT IN ENGINEERING SCIENCES

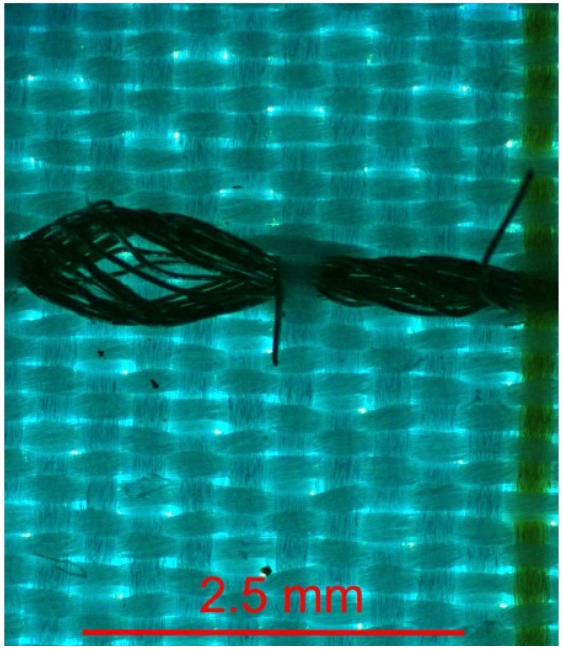
UNIVERSITÉ DU LUXEMBOURG

FACULTY OF SCIENCES, TECHNOLOGY AND COMMUNICATION

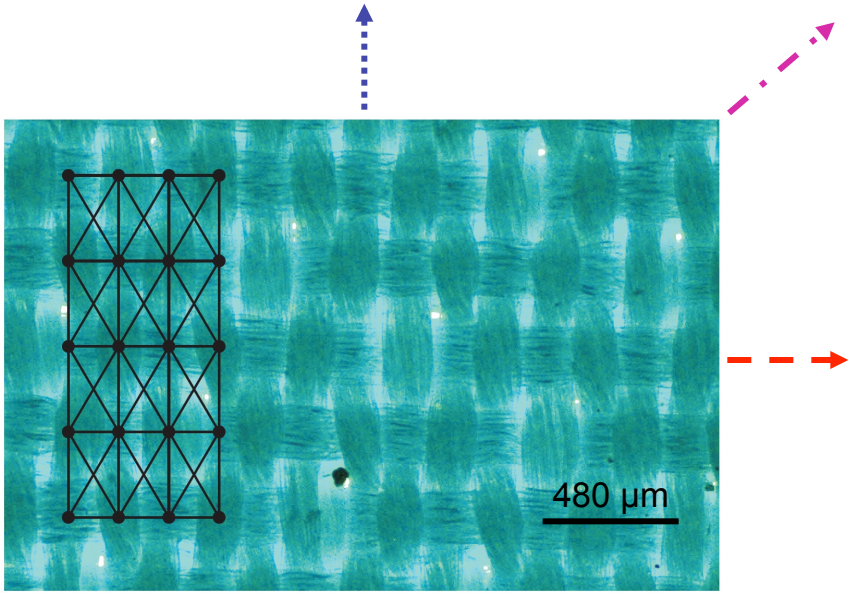
Introduction

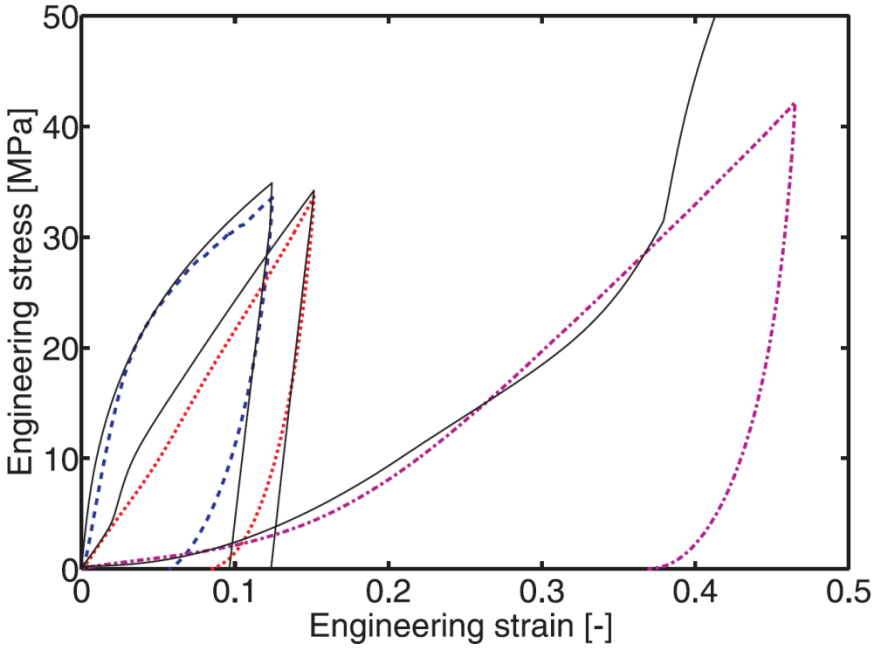




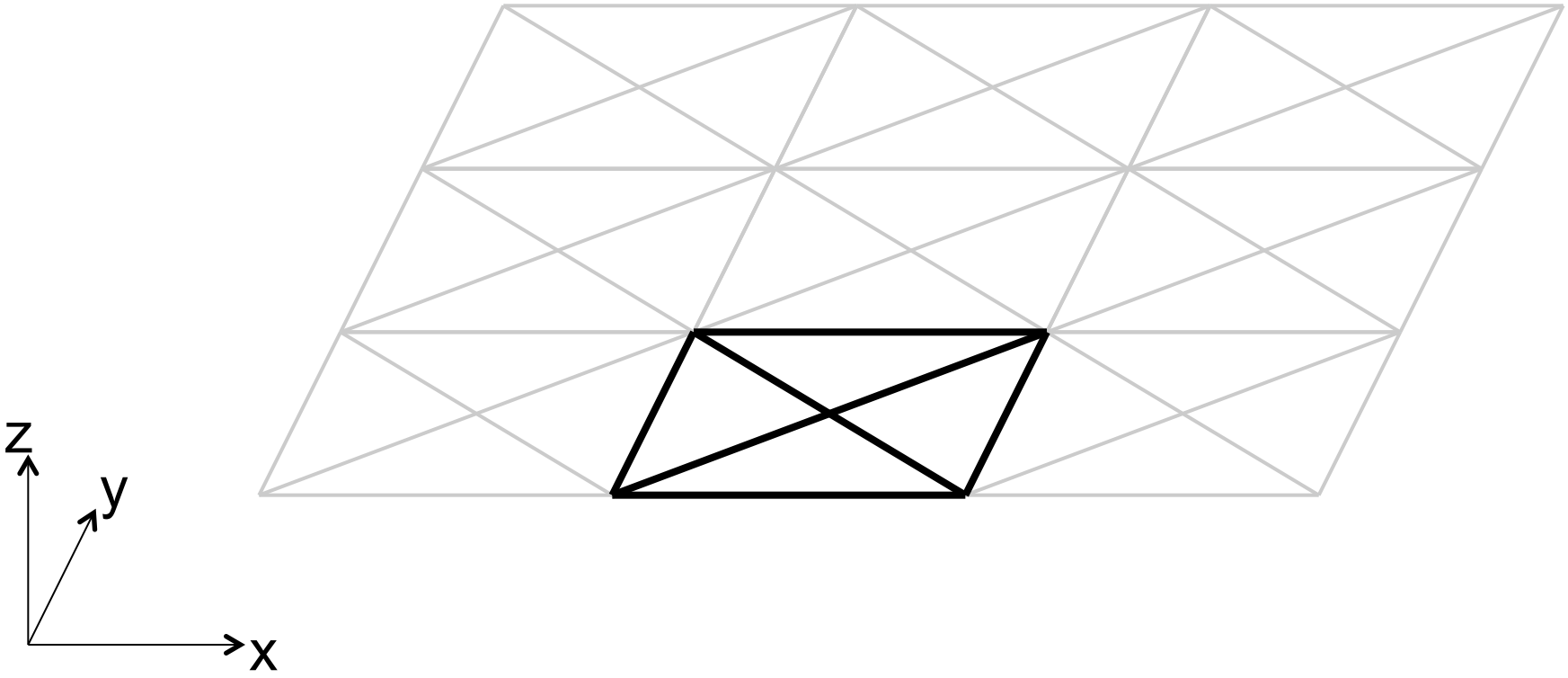


Introduction

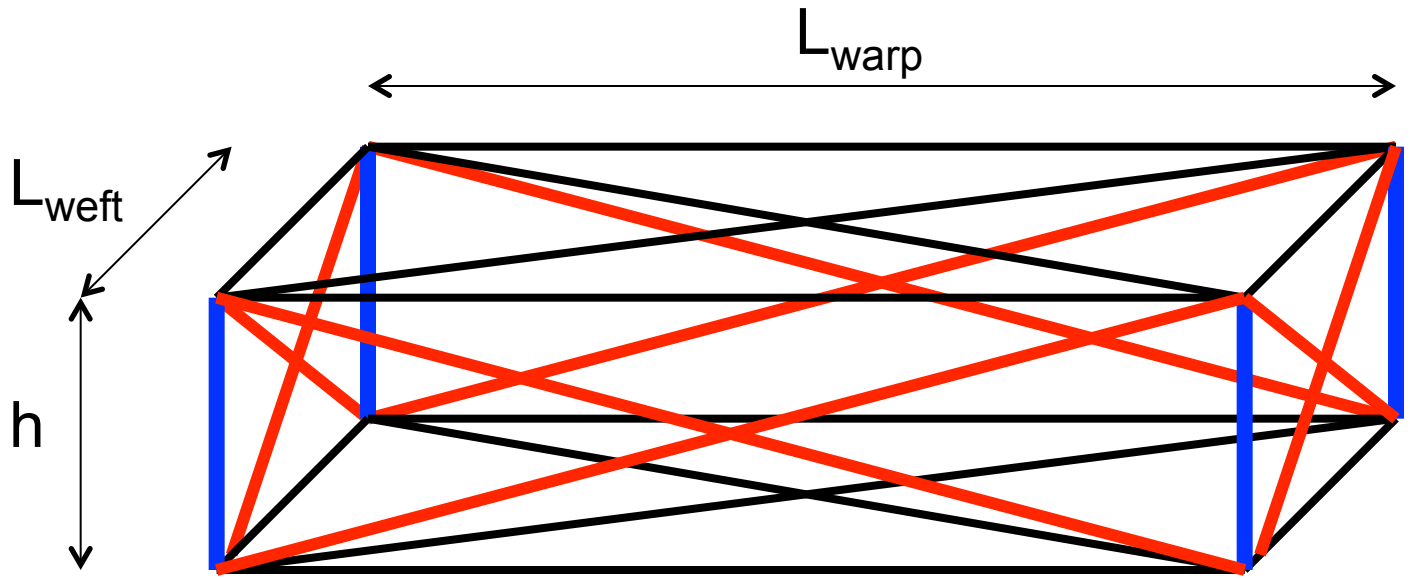


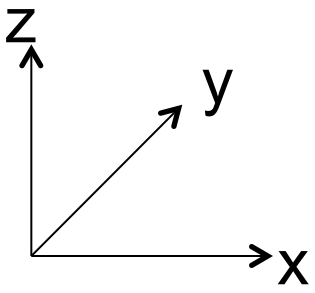


Problem: no out-of-plane stiffness



Creating out-of-plane stiffness: Approach 1

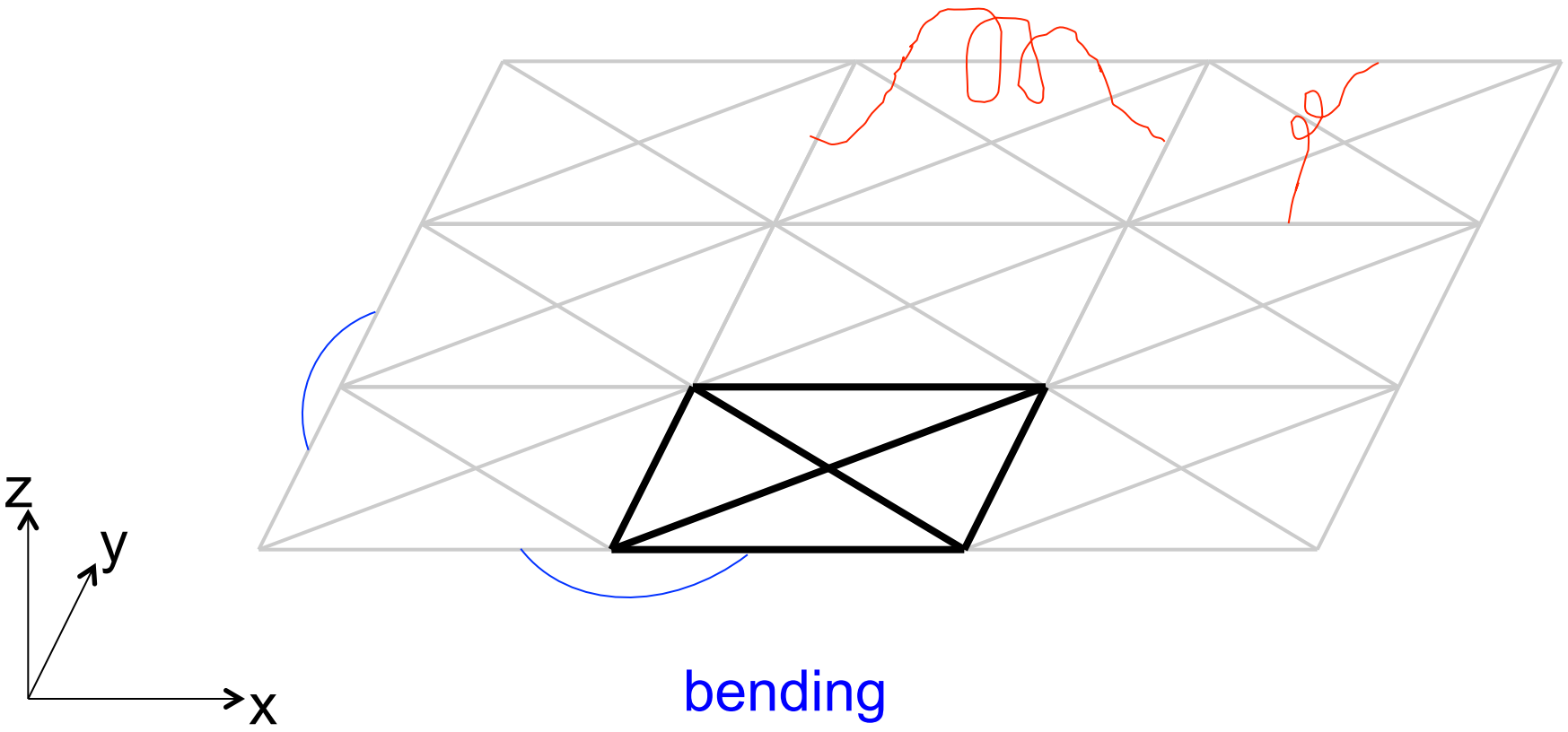




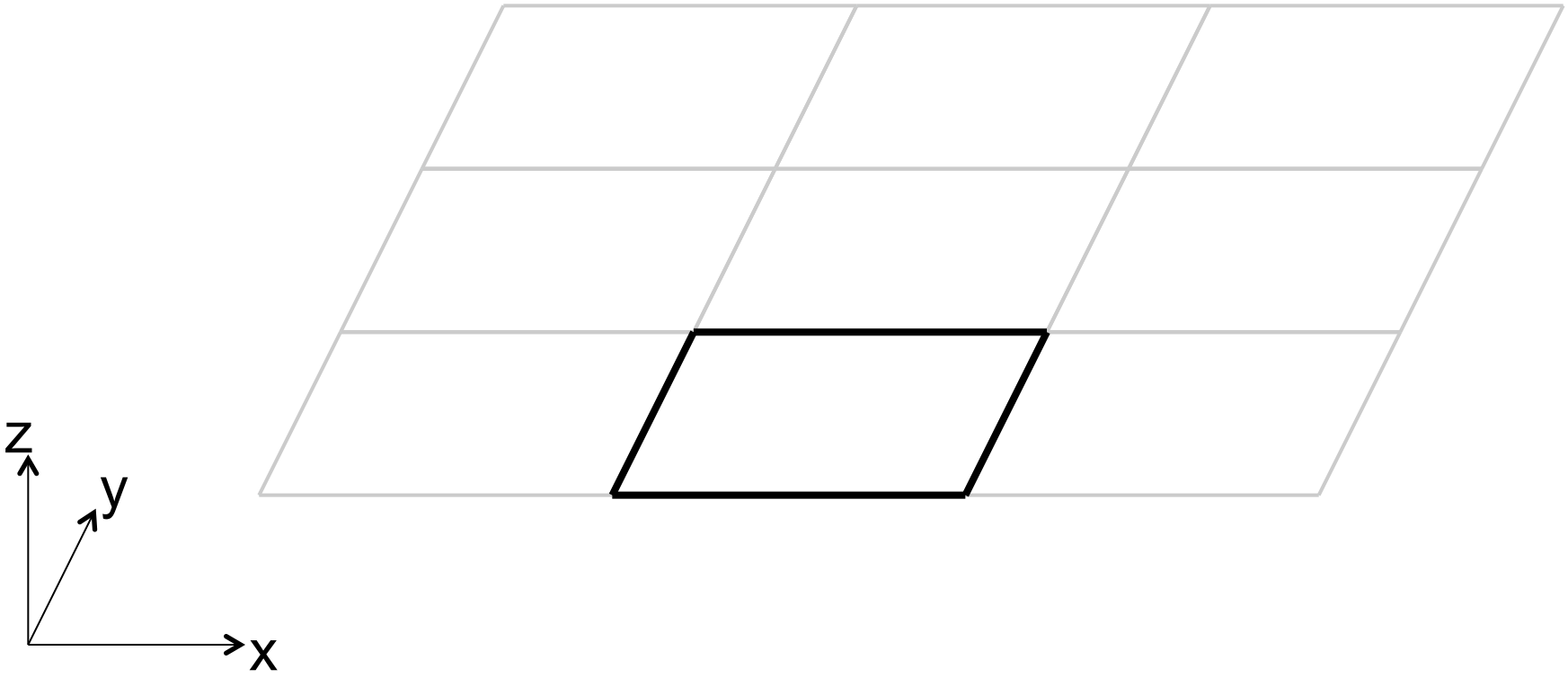
$h << L_{warp}$

$h << L_{weft}$

Creating out-of-plane stiffness: Approach 2



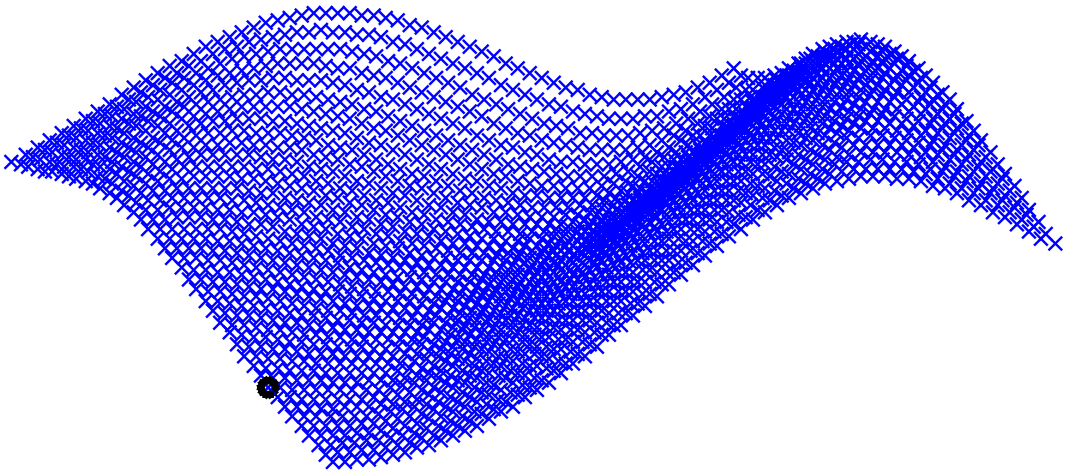
Energy minimisation without Hessian



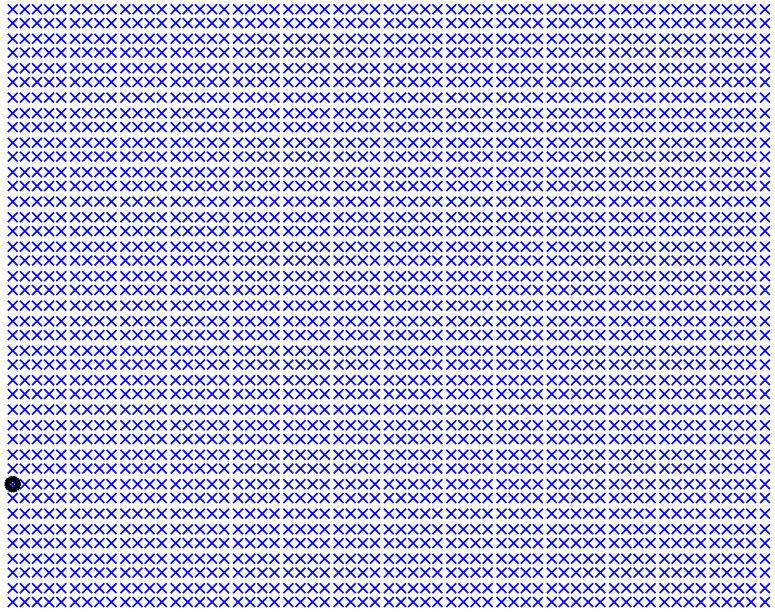
Energy minimisation: steepest descent

START: $\mathbf{u}_0$ $E(\mathbf{u}_0)$

View 1



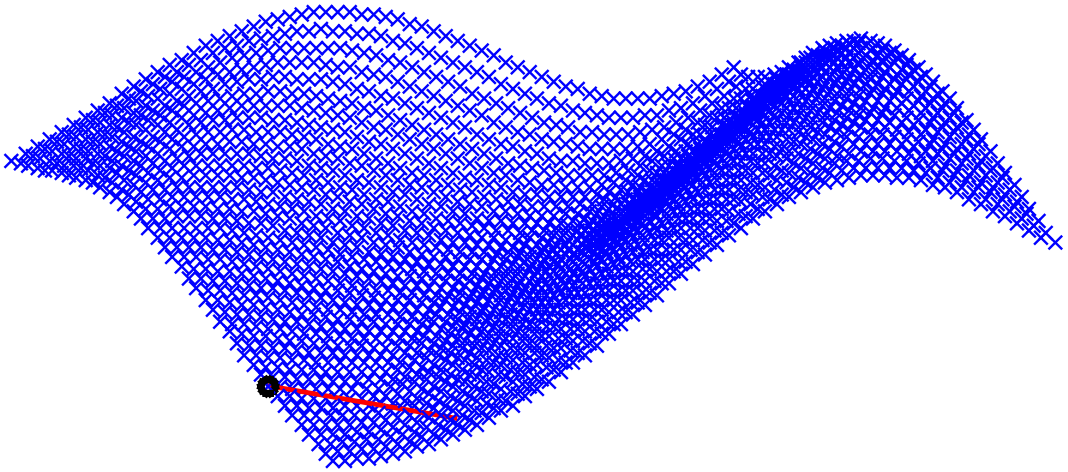
Top view



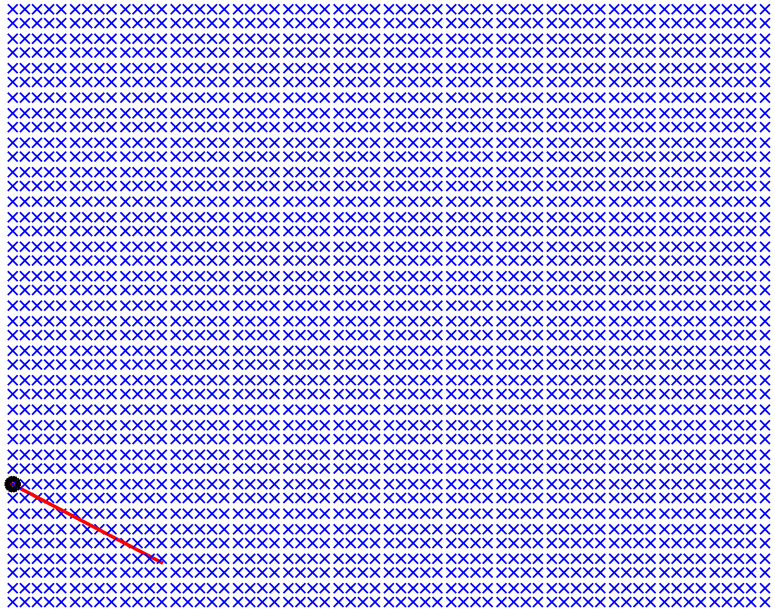
Energy minimisation: steepest descent

START: $\mathbf{u}_0$ $E(\mathbf{u}_0)$ $-\mathbf{f}(\mathbf{u}_0)$

View 1



Top view

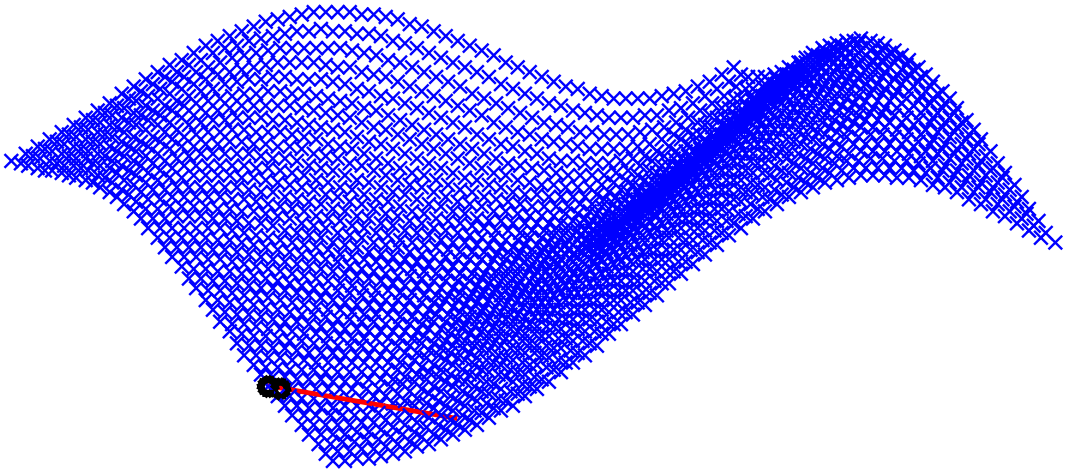


Energy minimisation: steepest descent

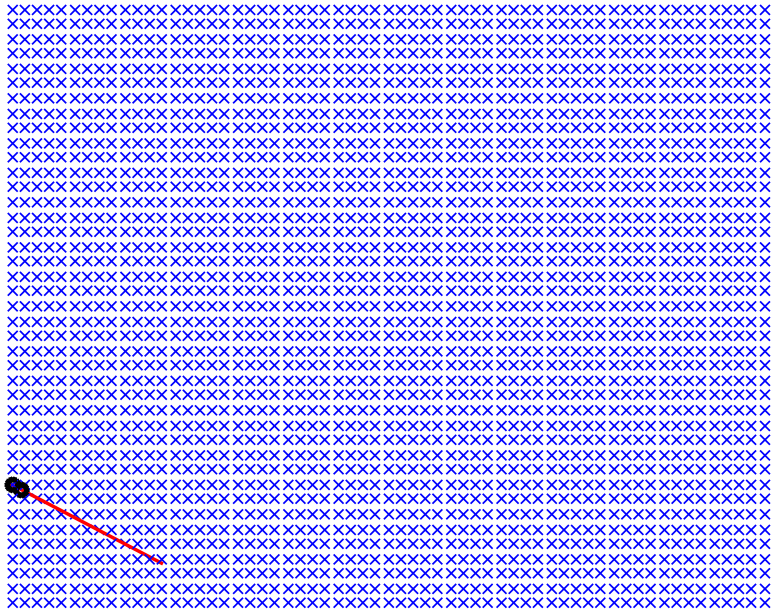
UPDATE: $\mathbf{u}_{i+1} = \mathbf{u}_i - \alpha\, \mathbf{f}(\mathbf{u}_i)$

$E(\mathbf{u}_{i+1})$

View 1



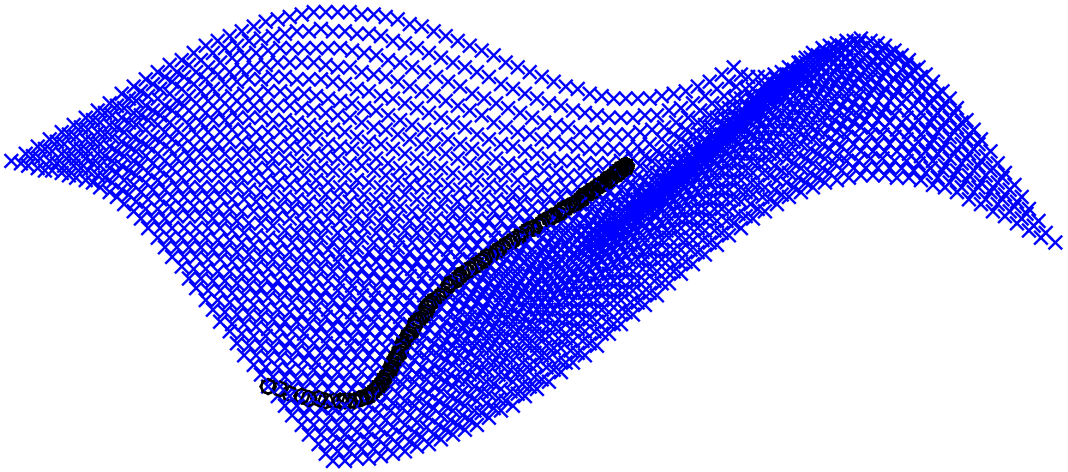
Top view



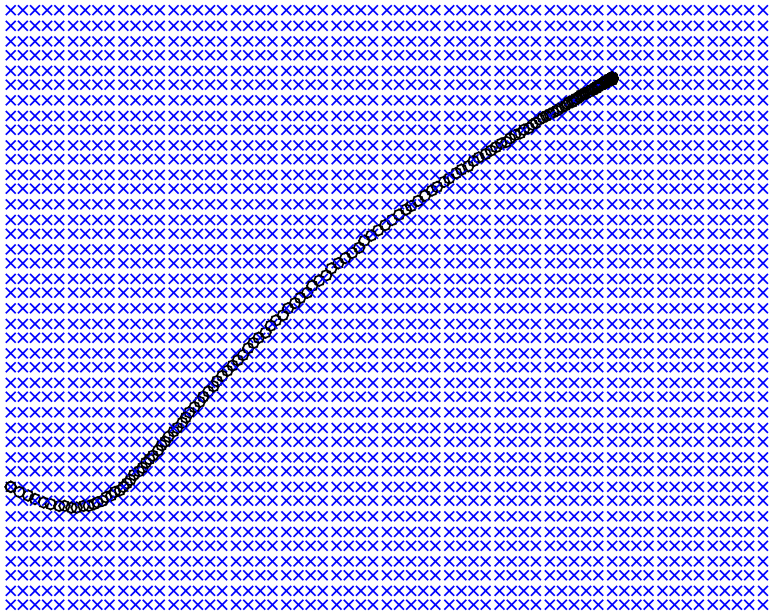
Energy minimisation: steepest descent

CONTINUE until $E(\mathbf{u}_{i+1}) > E(\mathbf{u}_i)$

View 1



Top view

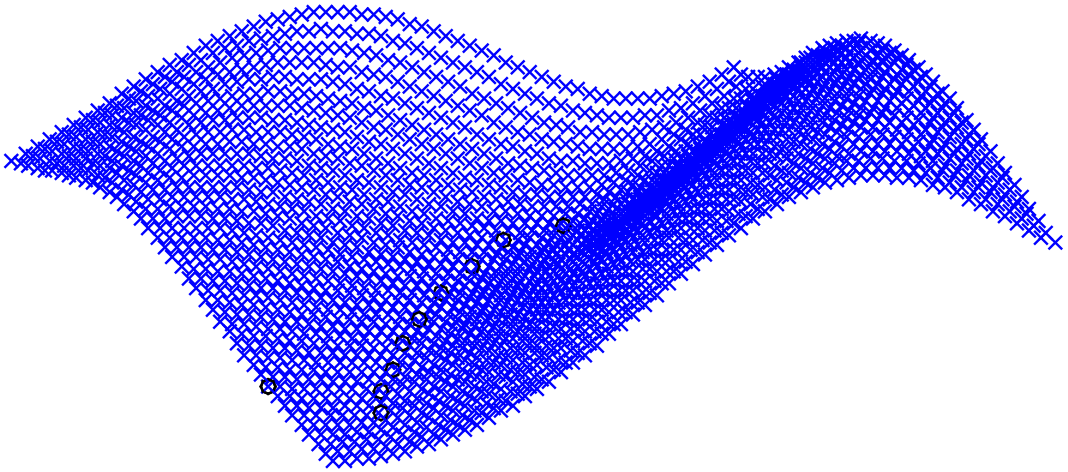


Energy minimisation: steepest descent

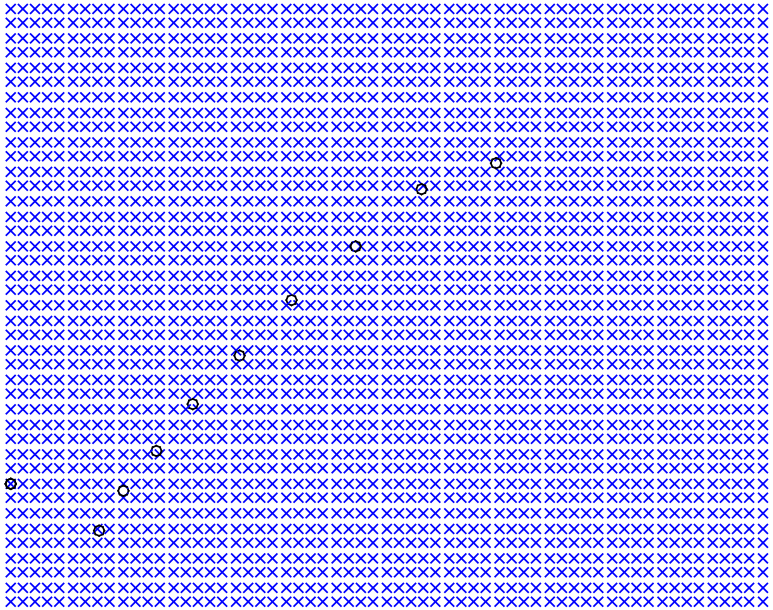
CONDITION: small stepsize α

→ this makes the procedure inefficient

View 1



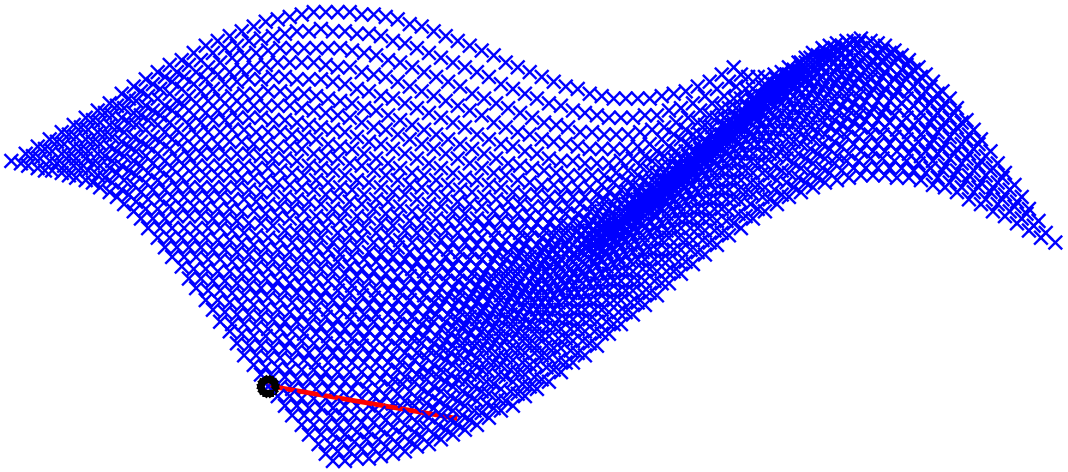
Top view



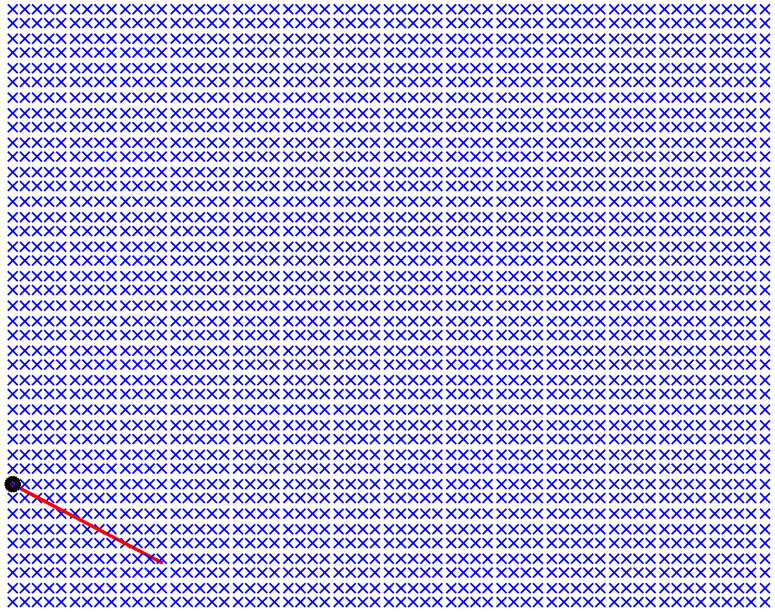
Steepest descent with backtracking linesearch (BTLS)

IDEA: once you know the search direction, optimise stepsize α such that $E(\mathbf{u}_{i+1})$ is minimum

View 1

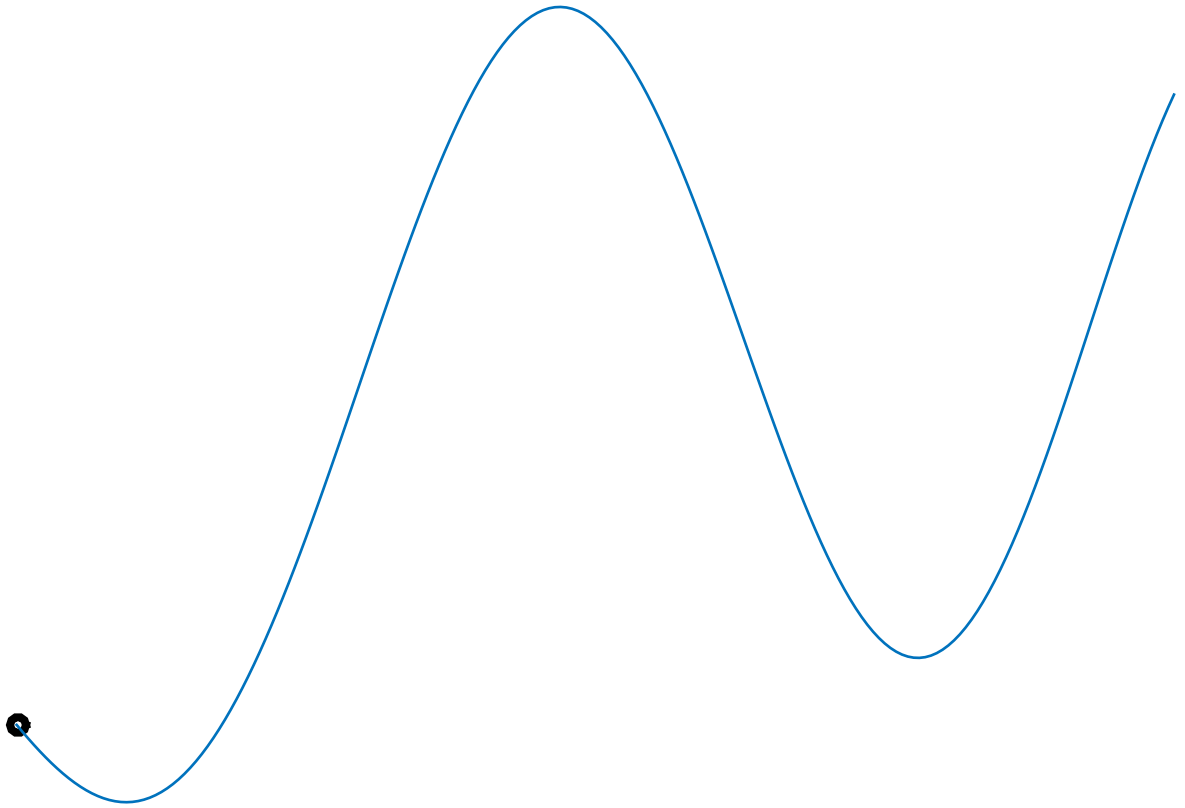


Top view



Steepest descent with backtracking linesearch (BTLS)

IDEA: once you know the direction, optimise stepsize α such that $E(\mathbf{u}_{i+1})$ is minimum



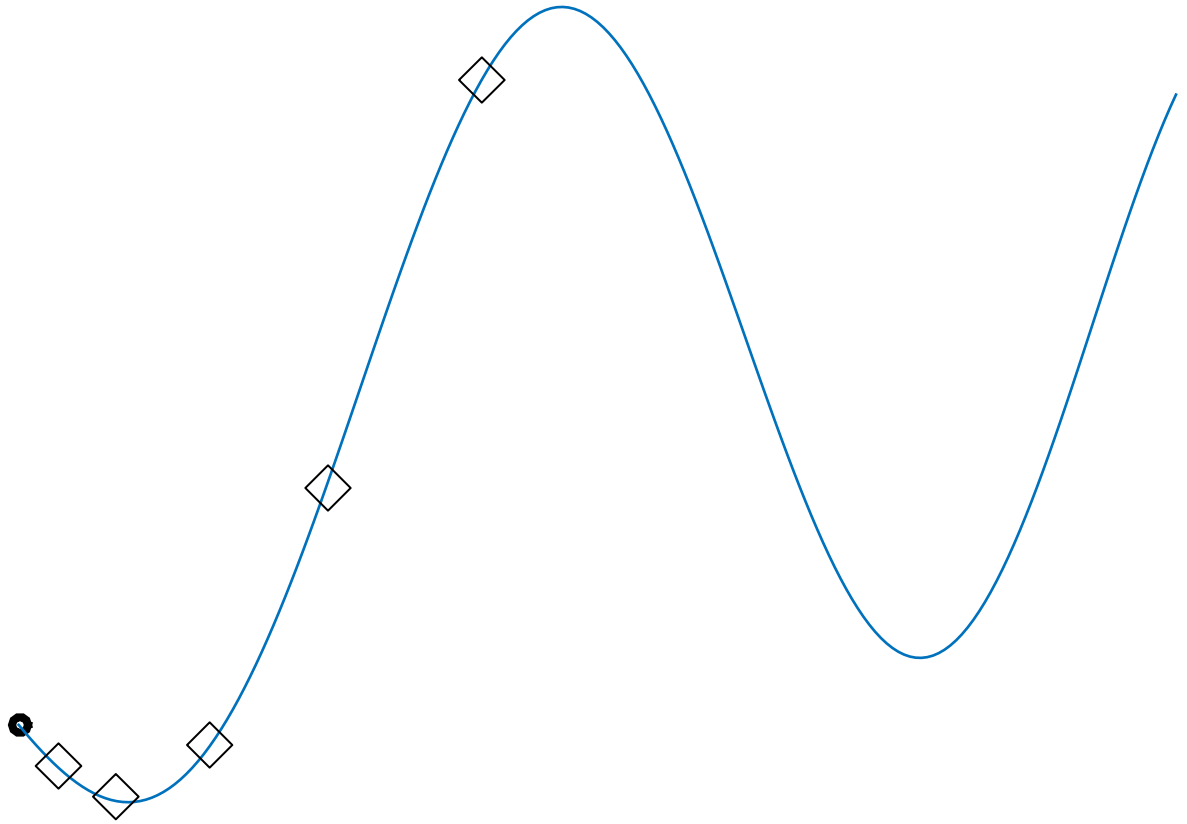
Cross-section along search direction

Steepest descent with backtracking linesearch (BTLS)

BACKTRACKING: $\alpha_i = \alpha_0$

while $E(\alpha_{i+1}) < E(\alpha_i)$

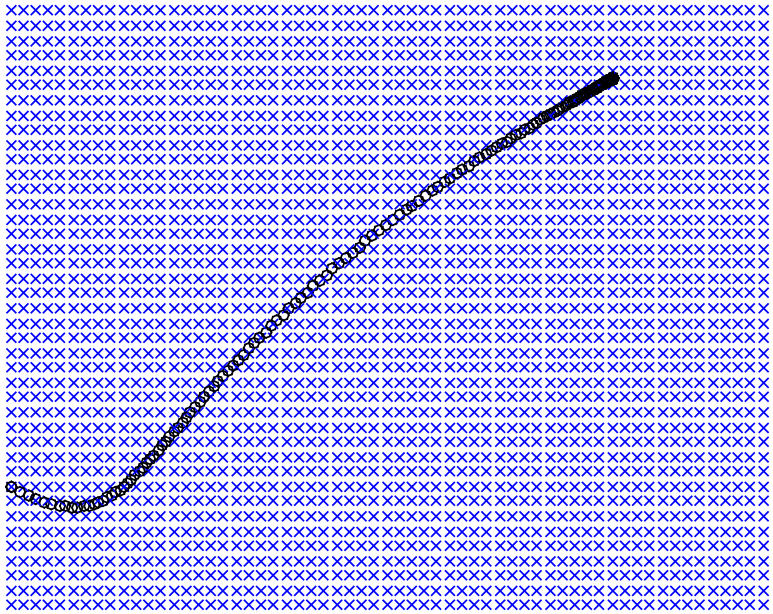
$\alpha_{i+1} = 0.5\ \alpha_i$



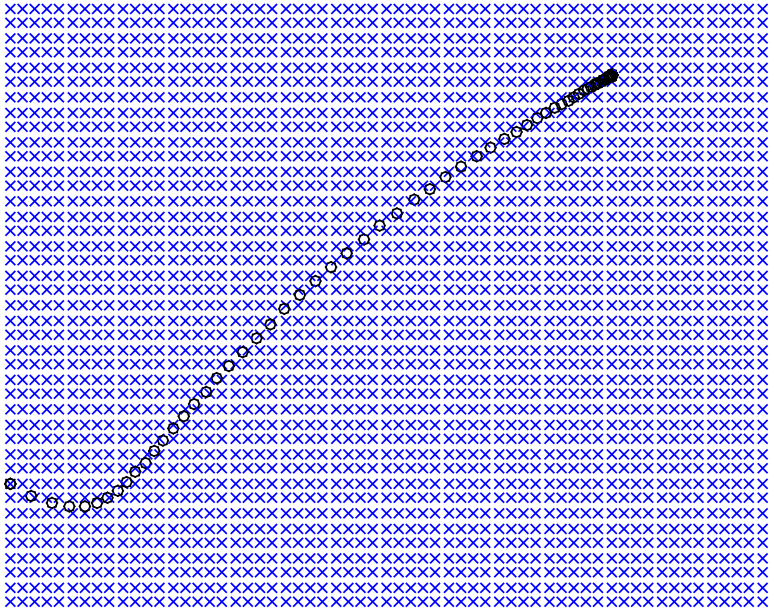
Cross-section along search direction

Steepest descent vs Steepest descent with BTLS

Steepest descent

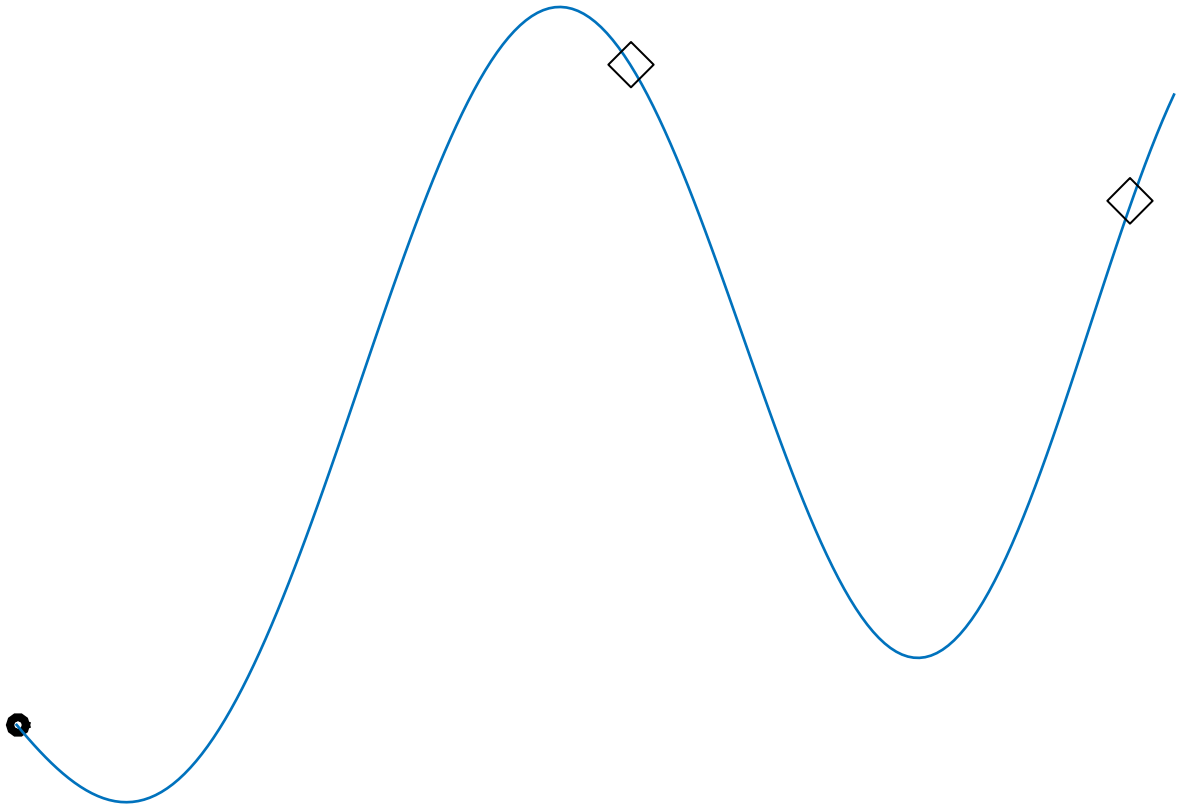


Steepest descent with BTLS



Steepest descent with backtracking linesearch (BTLS)

Problem BACKTRACKING: New point is higher than starting point

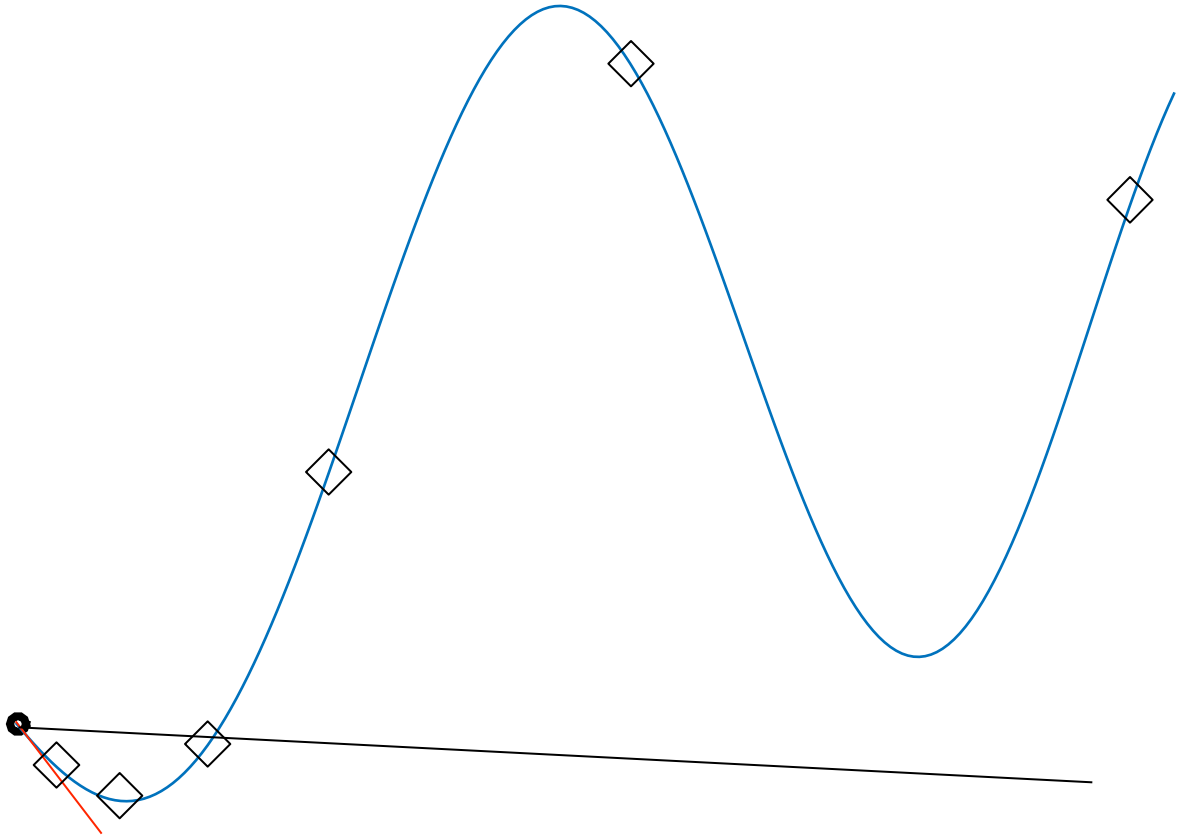


Cross-section along search direction

Steepest descent with BTLS and Armijo Rule (AR)

Solution: apply Armijo Rule, before backtracking

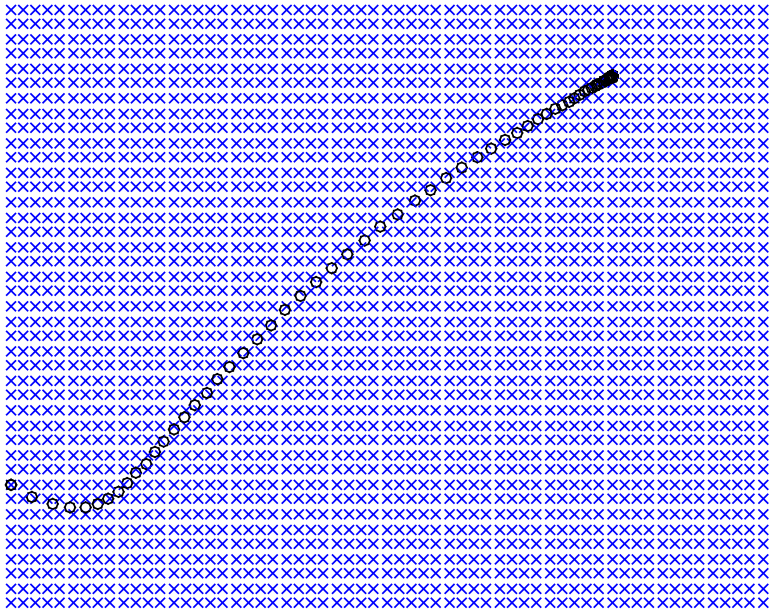
$$E(\mathbf{u}_i - \alpha_{i+1}\,\mathbf{f}(\mathbf{u}_i)) \leq E(\mathbf{u}_i) - c_1\,\alpha_0\,\mathbf{f}(\mathbf{u}_i)^T\,\mathbf{f}(\mathbf{u}_i)$$



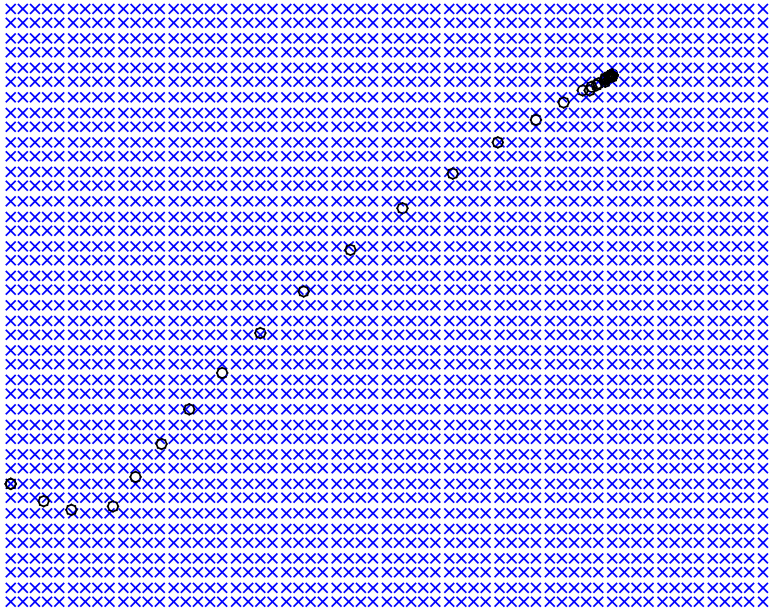
Cross-section along search direction

Steepest descent with BTLS vs Steepest descent with BTLS & AR

Steepest descent with BTLS

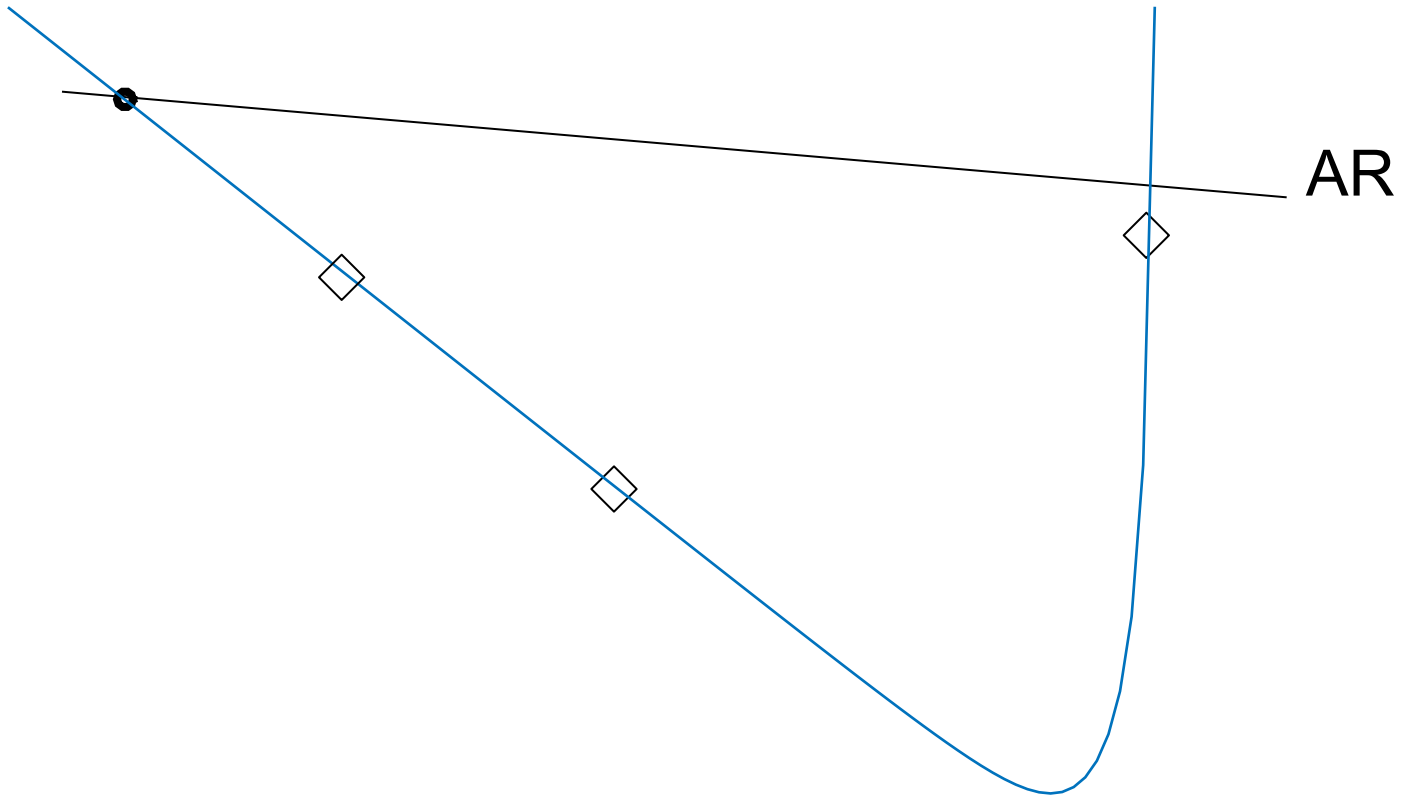


Steepest descent with BTLS & AR



Steepest descent with BTLS and Armijo Rule

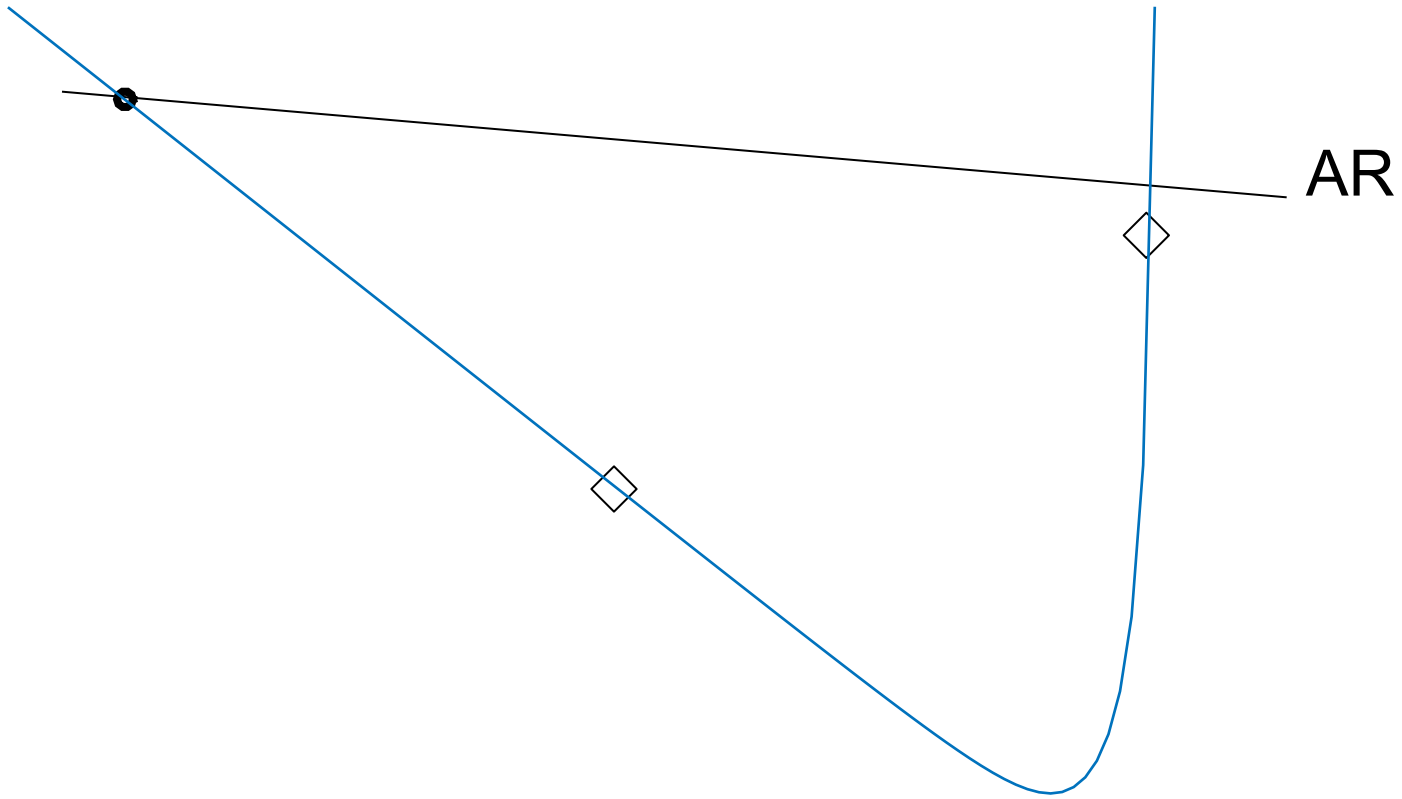
Consider the following case:



Cross-section along search direction

Steepest descent with BTLS and Armijo rule

Include Curvature Rule to ensure that the slope changes enough



Cross-section along search direction

Steepest descent with BTLS incl. AR and CR

Curvature Rule:

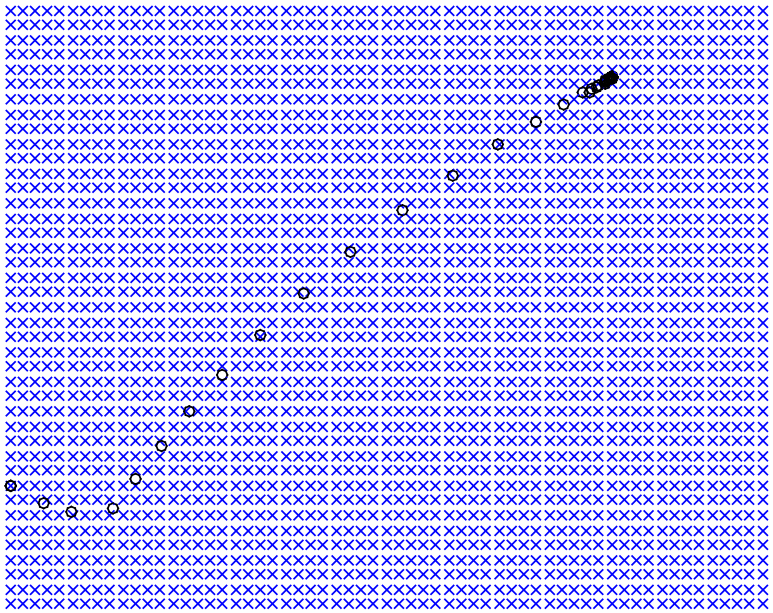
$$\mathbf{f}(\mathbf{u}_i)^T \mathbf{f}(\mathbf{u}_{i+1}) \leq c_2 \, \mathbf{f}(\mathbf{u}_i)^T \mathbf{f}(\mathbf{u}_i)$$

Terminology:

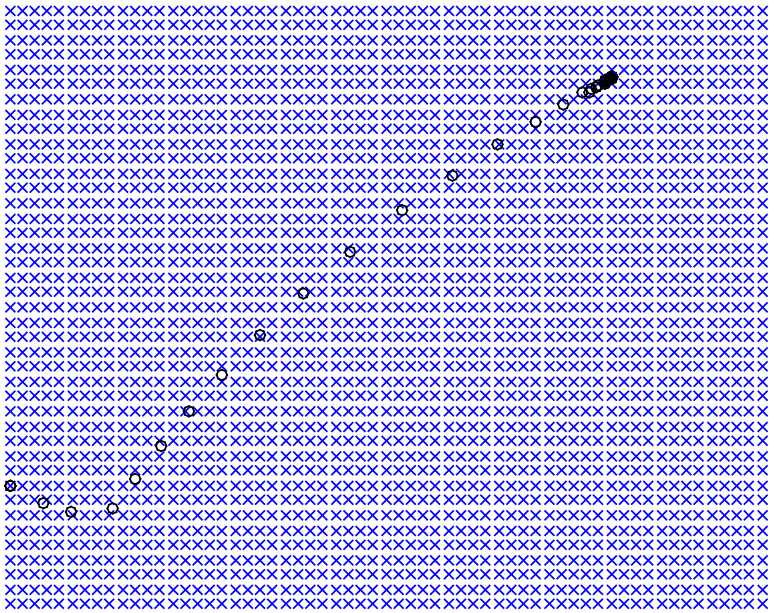
Armijo Rule + Curvature Rule = Wolfe conditions

Steepest descent with BTLS & AR vs Steepest descent with BTLS & WC

Steepest descent AR



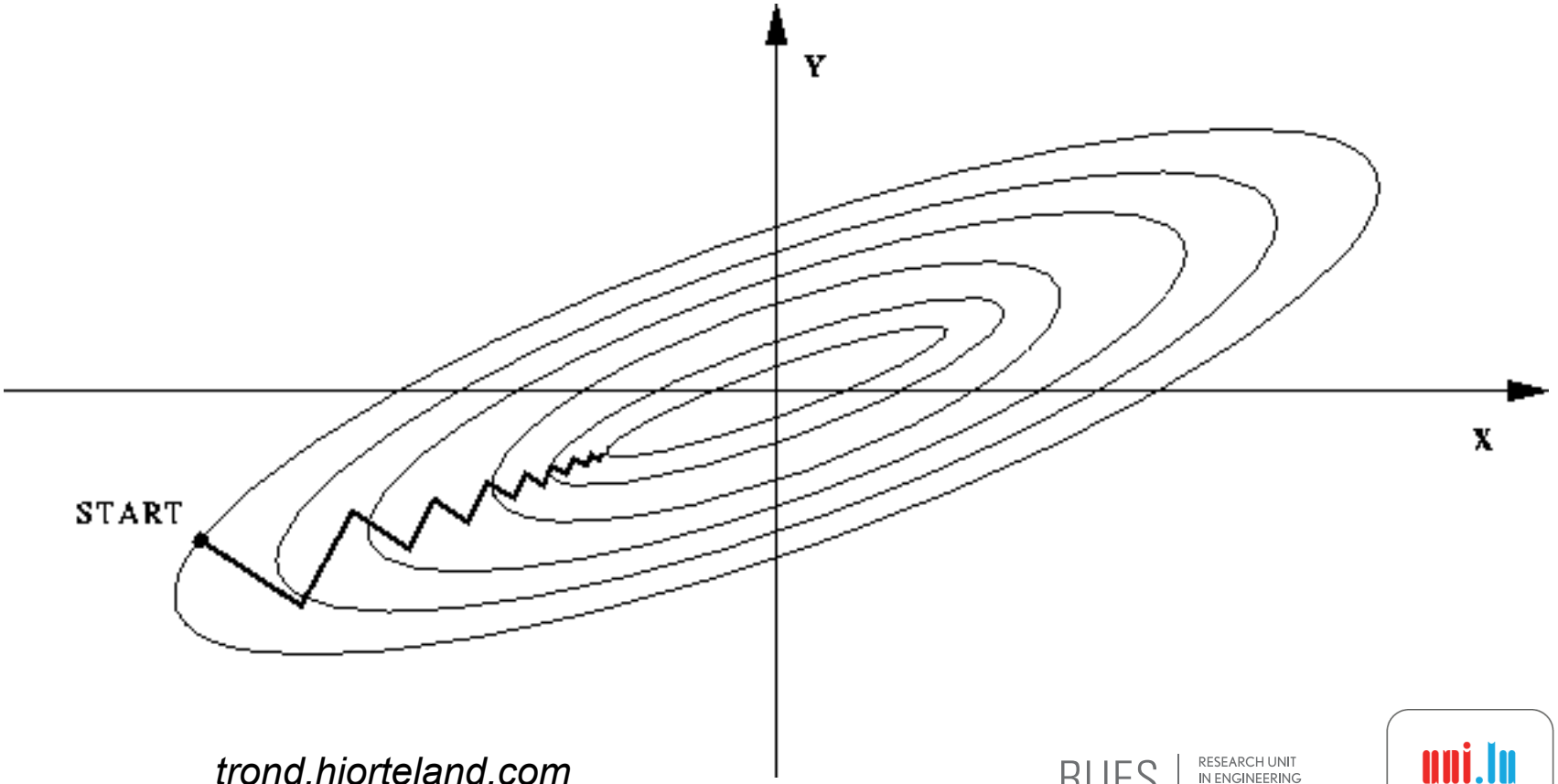
Steepest descent AR & CR



Conjugate gradient instead of Steepest descent

Enough about backtracking linesearch, consider this:

Combine current search direction with previous search direction



trond.hjorteland.com

Conjugate gradient vs Steepest descent

Steepest descent update:

$$\mathbf{u}_{i+1} = \mathbf{u}_i - \alpha\, \mathbf{f}(\mathbf{u}_i)$$

Conjugate gradient update for 2nd step:

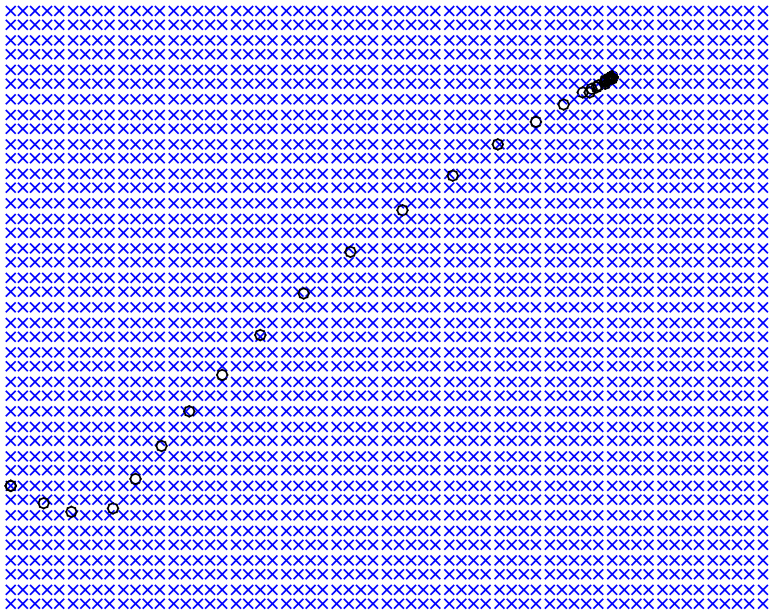
$$\mathbf{u}_2 = \mathbf{u}_1 - \alpha\, (\mathbf{f}(\mathbf{u}_1) - \beta\, \mathbf{f}(\mathbf{u}_0))$$

where β: $\beta(\mathbf{f}(\mathbf{u}_1), \mathbf{f}(\mathbf{u}_0))$

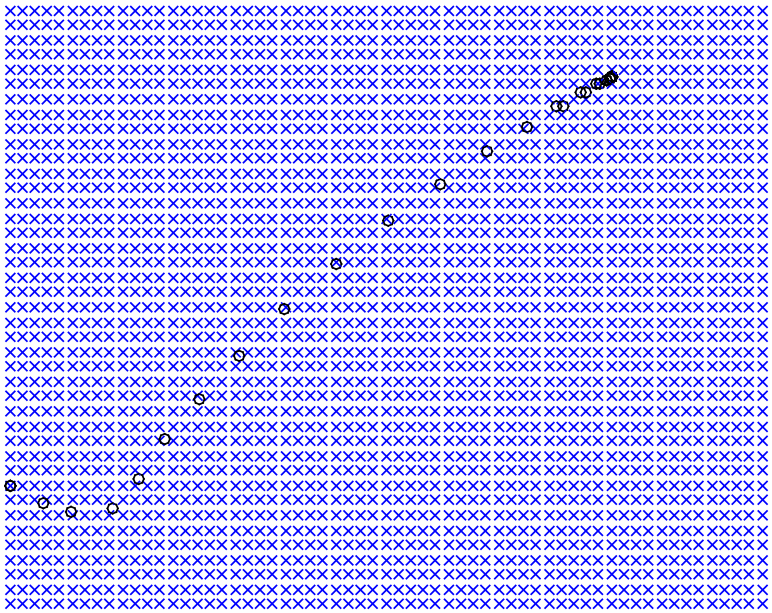
(e.g. Polak-Ribiere)

Steepest descent with BTLS & AR vs Steepest descent with BTLS & WC

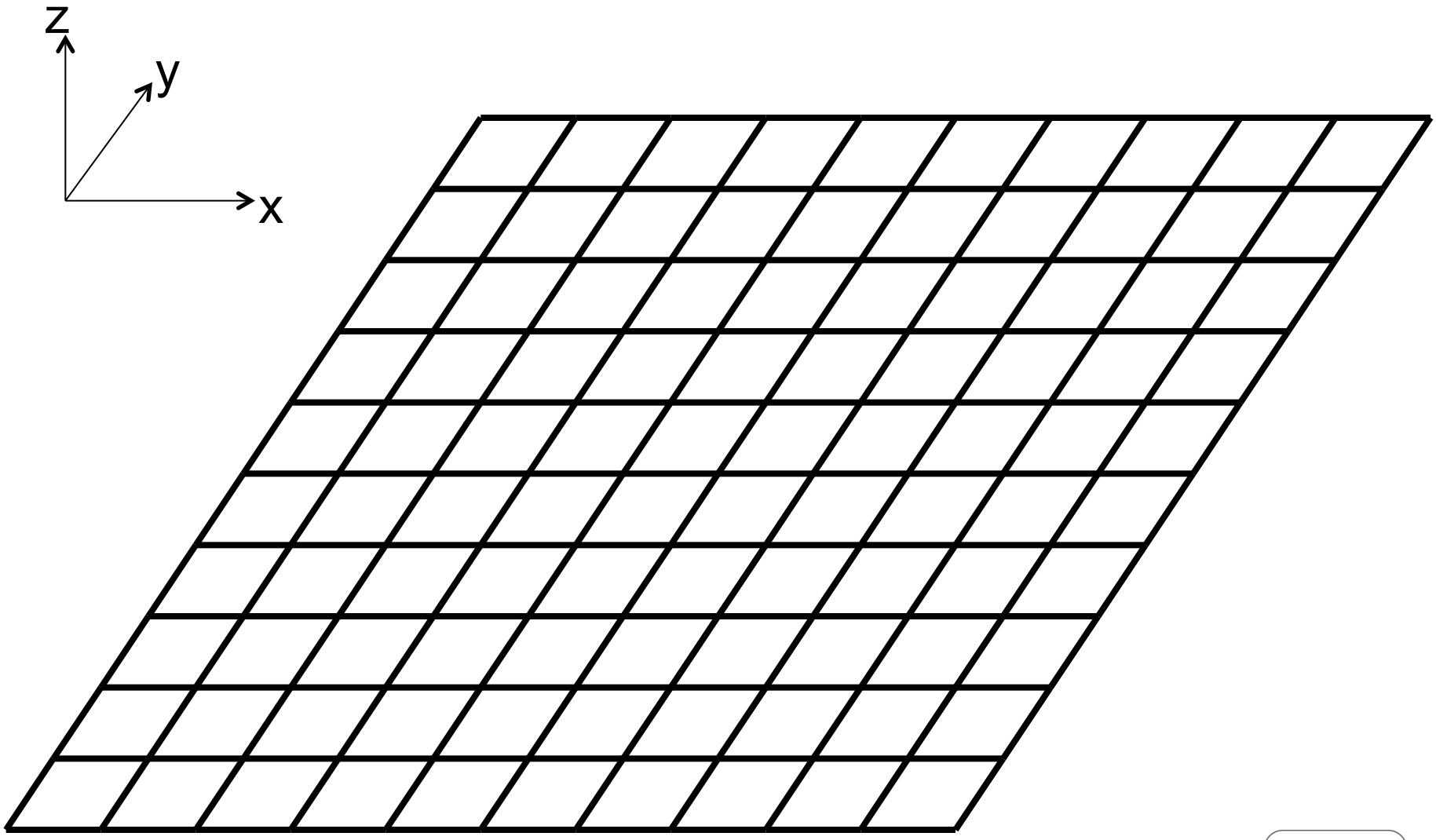
Steepest descent AR & CR



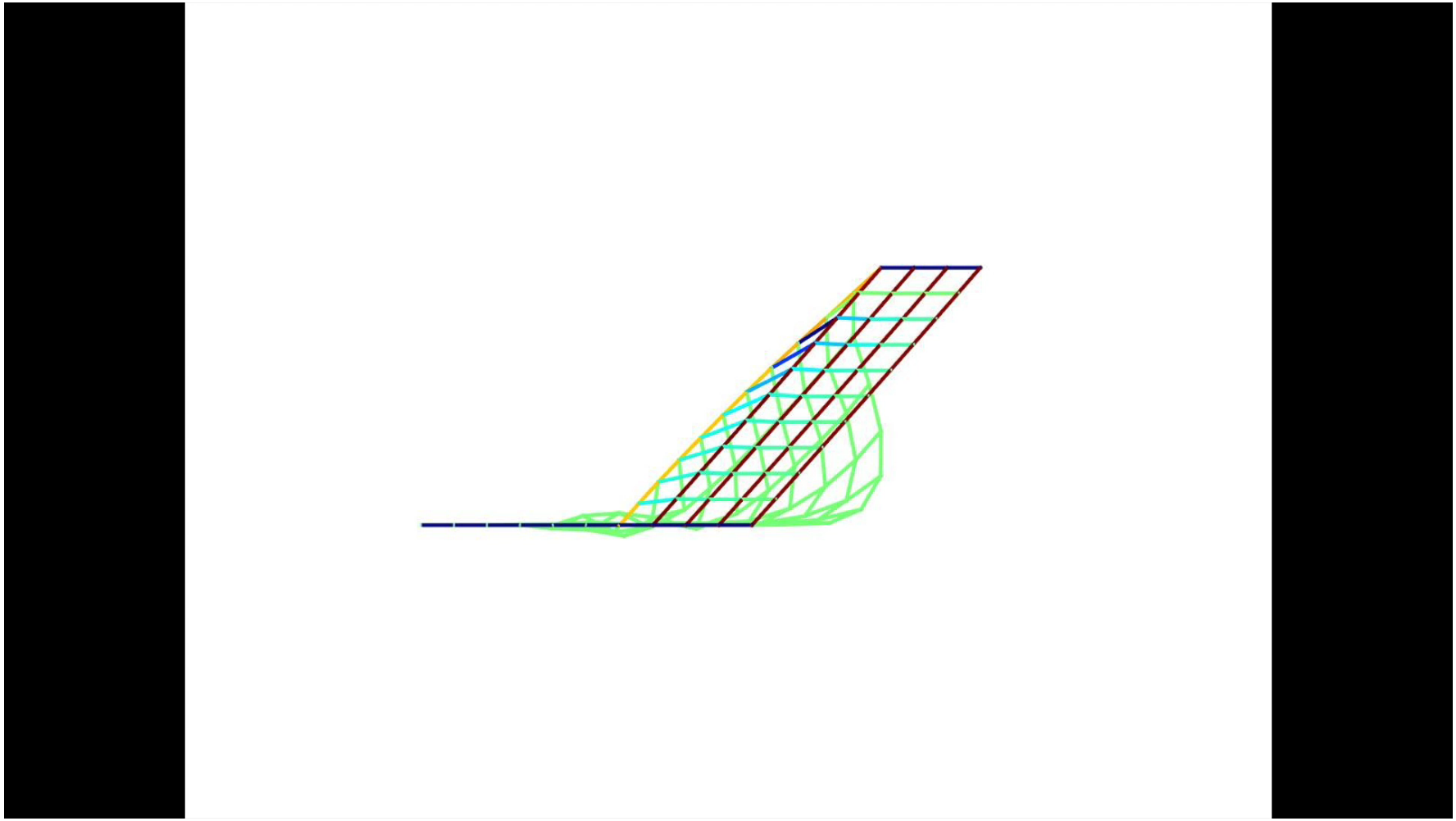
Conjugate gradient AR & CR



Truss network: Conjugate gradient with BTLS & WC



Truss network: Conjugate gradient with BTLS & WC



no scaling

Final remarks

Contact cannot be included via standard penalty methods or Lagrange multipliers

Energy landscape must be smooth

→ no discontinuous potential

→ no dissipation, BUT varational dissipation!!

→ no friction in contact

Alternative to avoid ill-posedness for truss structures:

→ beams